C++ Programming

(10154)

Rationale Statement:

The world is full of problems that need to be solved or that need a program to solve them faster. In computer, programming students will learn how to solve story problems and develop a computer program that will solve the problem. Computer programming courses in the state of South Dakota usually are taught using one of three computer languages: Visual Basic, C++, or Java. Students that are interested in the Programming and Software Development pathway will find that taking a computer-programming course in high school will better prepare them for post-secondary work in computer science, engineering, mathematics, and other software development areas.

Course Description:
Grade Level: 10-12

Course Topics:

- Introduction to programming history and the programming language
- Understanding the information processing cycle
- Customer needs analysis for designing a program
- Defining and designing the program project
- Coding an application
- Creating, debugging, and documenting a software application

NOTE: The core technical standards and examples are designed for a Programming 1 and Programming 2 Course.

Core Technical Standards & Examples

Indicator #1: Demonstrate programming as it relates to the customer needs	
Bloom's Taxonomy Level	**Standard and Examples**
Analyzing	**CCP 1.1 Gather data to identify customer requirements.** Examples: • Gather information using interviewing strategies. • Identify input, output and system processing requirements. • Clarify specifications using questioning techniques. • Identify hardware, networking, and software system functional requirements. • Demonstrate knowledge of nonfunctional requirements. • Use customer satisfaction in determining product characteristics.
Applying	**CCP 1.2 Demonstrate knowledge of programming language concepts.** Examples: • Demonstrate knowledge of the concept of physical representation of digitized information. • Demonstrate knowledge of the hardware-software connection. • Demonstrate knowledge of the function and operation of compilers and interpreters. • Demonstrate knowledge of current key programming languages and the environment they are used in.
Evaluating	**CCP 1.3 Develop software requirements specification.** Examples: • Demonstrate knowledge of the use, structure, and contents of a requirements specification document. • Define system and software requirements. • Define business problem to be solved by the application • Develop informal specifications. • Develop formal specification. • Review and verify specification with customer.

Indicator #2: Produce IT-based strategies and project plans to solve the problem.	
Bloom’s Taxonomy Level	**Standard and Examples**
Understanding	**CCP 2.1 Define scope of work for the programming project.** Examples: • Demonstrate knowledge of the key functions and subsystems of the software product. • Demonstrate knowledge of software development process and issues. • Develop implementation plan.
Applying	**CCP 2.2 Demonstrate knowledge and skills of working in a software development team.** Examples: • Identify resources and risks. • Demonstrate knowledge of cross-functional team structures and team members’ roles.

Indicator #3: Demonstrate knowledge of the software development process.	
Bloom’s Taxonomy Level	**Standard and Examples**
Understanding	**CCP 3.1 demonstrate knowledge of software development methodology.** Examples: • Demonstrate knowledge of system analysis issues related to design, testing, implementation, and maintenance. • Identify roles on team members/customers in the software development process. • Demonstrate knowledge of how to use software methodologies to analyze a real-world problem. • Identify constraints of the current project. • Demonstrate knowledge of modeling and analyzing functional requirements (e.g., dataflow diagrams, process specifications, and a data dictionary). • Demonstrate knowledge of modeling and analyzing data requirements (e.g., Jackson diagrams, entity relationship diagrams, and relations).
Applying	**CCP 3.2 Apply tools for developing software applications.** Examples: • Demonstrate knowledge of software development environment. • Use prototyping techniques. • Use desk checking • Demonstrate knowledge of reuse and components.
Applying	**CCP 3.3 Apply language specific programming tools/techniques.** Examples: • Develop programs using appropriate language. • Make use of appropriate development environment for the selected language.

Indicator #4: Create a logical design for a software application.	
Bloom's Taxonomy Level	**Standard and Examples**
Evaluating	**CCP 4.1 Create design specification for a computer application.** Examples: • Analyze real world problems for the applicability of structured, object oriented, event-driven logical design methods. • Design system input, output, processing, and interfaces.
Applying	**CCP 4.2 Analyze real world problems for the applicability of structured, object orientate, even driven logical design methods.** Examples: • Demonstrate knowledge of the characteristics and the uses of processing • Identify basic concepts of algorithm and data structure development. • Demonstrate knowledge of different data types • Identify constraints. • Demonstrate knowledge of nonfunctional requirements • Demonstrate knowledge of modular design concepts. • Demonstrate knowledge of how to design and implement programs in a top-down manner. • Analyze and prepare logic using program flowchart. • Identify standards and issues related to I/O programming and design of I/O interfaces.

Indicator #5: Create a computer application by writing code.	
Bloom's Taxonomy Level	**Standard and Examples**
Applying	**CCP 5.1 Demonstrate knowledge of programming language concepts.** Examples: • Demonstrate knowledge of the basics of structured, object-oriented, and event-driven programming. • Demonstrate knowledge of the concepts of data and procedural representation.
Applying	**CCP 5.2 Develop an application using selected programming language.** Examples: • Translate logical design into code in an appropriate language argument. • Demonstrate knowledge of specific language syntax.
Evaluating	**CCP 5.3 Demonstrate knowledge of basic software systems implementation.** Examples: • Compile and debug code. • Prepare code documentation. • Conduct code walkthrough and/or inspection. • Troubleshoot unexpected results. • Access needed information using company and manufacturers' references.